## G. Les enregistrements (le type RECORD)

**Définition : *RECORD***

Un ***RECORD*** ou enregistrement est une structure de données (similaire au type ***struct*** en langage C) qui peut contenir des valeurs de types différents (NUMBER, DATE, VARCHAR2, …).

Il existe 3 types d'enregistrements :

- Les enregistrements basés sur la structure d'une table
- Les enregistrements basés sur la structure d'un curseur
- Les enregistrements dont la structure est définie par le programmeur

***Les enregistrements basés sur la structure d'une table***

Syntaxe de déclaration (bloc **DECLARE**):

```
nom_record "nom_table"%ROWTYPE;
```

Syntaxe d'initialisation :

```
SELECT * into nom_record
FROM "nom_table"
WHERE condition;
```

La requête de sélection doit obligatoirement retourner **une seule ligne**

Syntaxe d'utilisation :

```
nom_record.nom_colonne
```

***Exemple***

1. On se propose d'abord d'exécuter la requête (**CREATE TABLE**) de création de la table « **_produit** » ayant la structure suivante :

| Name | Type | Length | Scale | Allow Null | |
|---|---|---|---|---|---|
| id | NUMBER | 3 | 0 | ☐ | 1 |
| designation | VARCHAR2 | 100 | 0 | ☐ | |
| qteStock | NUMBER | 2 | 0 | ☐ | |
| prixUnit | NUMBER | 6 | 2 | ☐ | |

2. La table sera ensuite remplie des données suivantes (**INSERT INTO**) :

| id | designation | qteStock | prixUnit |
|---|---|---|---|
| 556 | D. DUR Ext. Verbatim VRB3360V2 500GB | 11 | 990,95 |
| 555 | Clef USB HP 10V02-16GB | 2 | 220 |

3. On utilise la structure de cette table pour créer et utiliser un enregistrement, qui va contenir les informations du produit '556'. Afficher ensuite ces informations.

```
DECLARE
   produit_rec "_produit"%ROWTYPE;
BEGIN
   SELECT * into produit_rec FROM "_produit" WHERE id = 556;

   dbms_output.put_line('ID produit: ' || produit_rec.id);
   dbms_output.put_line('Désignation: ' || produit_rec.designation);
   dbms_output.put_line('Qté stock: ' || produit_rec.qteStock);
   dbms_output.put_line('Prix unitaire: ' || produit_rec.prixUnit);
END;
```

```
Message | DBMS Output
ID produit: 556
Désignation: D. DUR Ext. Verbatim VRB3360V2 500GB
Qté stock: 11
Prix unitaire: 990.95
```


**IMPORTANT**

Un enregistrement ne peut contenir qu'une seule ligne de la table, donc la clause **WHERE** est **obligatoire** si la table contient plusieurs lignes, et il faut que la requête **SELECT** retourne **une seule ligne**.
Si la table ne contient qu'une seule ligne, la clause **WHERE n'**est pas obligatoire.

### *Les enregistrements basés sur la structure d'un curseur*

Syntaxe de déclaration (bloc **DECLARE**) :

```
/* déclaration du curseur */
......
/* déclaration du RECORD */
nom_record "nom_curseur"%ROWTYPE;
```

Syntaxe d'utilisation :

```
/* ouverture du curseur */
......
nom_record."nom_colonne"
```

*(Les notions de curseur, et curseur avec enregistrement seront revues en détail dans les prochains chapitres)*

### *Les enregistrements basés sur une structure définie par le programmeur*

Syntaxe de déclaration (bloc **DECLARE**) :

Optionnel

Optionnel, mais obligatoire si **NOT NULL**

```
/* définition du RECORD */
TYPE nom_record IS RECORD (
      Attribut_1 TYPE_DE_DONNEES [NOT NULL] [ := VALEUR],
      Attribut_2 TYPE_DE_DONNEES [NOT NULL] [ := VALEUR],
      ......
      Attribut_N TYPE_DE_DONNEES [NOT NULL] [ := VALEUR]
);

/* déclaration d'un RECORD */
nom_variable nom_record;
```

Syntaxe d'utilisation :

```
nom_variable."nom_attribut"
```

### *Exemple 1*

On se propose de refaire l'exemple précédent, **sans** utiliser la table « **_produit** » comme structure de base pour la création du record.

```
DECLARE
      /* définition du type RECORD nommé : "product" */
      TYPE product IS RECORD (
            id NUMBER NOT NULL := 1,
            designation VARCHAR2(50),
```

```
            qteStock NUMBER,
            prixUnit NUMBER(6,2)
      );

      /* déclaration d'un product nommé produit_1 */
      produit_1 product;
BEGIN
      /* initialisation des attributs du product produit_1 */
      produit_1.designation := 'PC Portable';
      produit_1.qteStock := 5;
      produit_1.prixUnit := 7000;
      dbms_output.put_line('ID produit: ' || produit_1.id);
      dbms_output.put_line('Désignation: ' || produit_1.designation);
      dbms_output.put_line('Qté stock: ' || produit_1.qteStock);
      dbms_output.put_line('Prix unitaire: ' || produit_1.prixUnit);
END;
```

Message | DBMS Output

```
ID produit: 1
Désignation: PC Portable
Qté stock: 5
Prix unitaire: 7000
```

**Exemple 2**

On se propose encore de refaire l'exemple précédent, **sans** utiliser la table « **_produit** » comme structure de base, **mais en utilisant ses types de colonnes pour typer les attributs du record (*%TYPE*)**.

```
DECLARE
      /* définition du type RECORD nommé : "product" */
      TYPE product IS RECORD (
            id "_produit".id%TYPE NOT NULL := 1,
            designation "_produit".designation%TYPE,
            qteStock "_produit".qteStock%TYPE,
            prixUnit "_produit".prixUnit%TYPE
      );

      /* déclaration d'un product nommé produit_1 */
      produit_1 product;
BEGIN
      /* initialisation des attributs du product produit_1 */
      produit_1.designation := 'PC Portable';
      produit_1.qteStock := 5;
      produit_1.prixUnit := 7000;
      dbms_output.put_line('ID produit: ' || produit_1.id);
      dbms_output.put_line('Désignation: ' || produit_1.designation);
      dbms_output.put_line('Qté stock: ' || produit_1.qteStock);
      dbms_output.put_line('Prix unitaire: ' || produit_1.prixUnit);
END;
```

***Etude de cas***

***Records – table « commande »***

1. Créer la table «_**commande** » ayant la structure suivante :

| Name | Type | Length | Scale | Allow Null | |
|---|---|---|---|---|---|
| NUMCOM | NUMBER | 4 | 0 | ☐ | 1 |
| DATECOM | DATE | 7 | 0 | ☐ | |

2. La table sera ensuite remplie des données suivantes :

| NUMCOM | DATECOM |
|---|---|
| 11 | 2010-02-19 |
| 12 | 2010-02-22 |
| 14 | 2010-03-01 |
| 16 | 2012-04-10 |

3. Utiliser un enregistrement pour afficher les données **de la dernière ligne de la table**.